Problem A. Nutella's Life

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Website `chefforces.at` has just published a schedule of contests for the next year! There will be n contests and no changes in the schedule. Oleg got really excited and decided to maximize the fun.

After a thorough analysis of each contest's problem setters Oleg came up with n integers a_i, one for each contest. The number a_i is the amount of fun Oleg will get while playing the i-th contest. Note that, due to *a notorious coincidence*, some numbers a_i can be negative.

However, Oleg does not want to miss contests and, especially, to miss several contests in a row. Formally, if Oleg decides to skip a contest, and he has already skipped x contests which took place immediately before this one, his total fun decreases by $x + 1$.

Finally, Oleg wants each contest to be at least as fun as the previous one in which he has participated. In other words, if Oleg participates in contests with numbers i and j, $i < j$, then the condition $a_i \leq a_j$ must hold.

Help Oleg to decide in which contests he has to participate in order to maximize the total fun.

Input

The first line contains an integer n, the number of contests in the schedule ($1 \leq n \leq 10^5$).

The second line contains n integers a_i ($-10^9 \leq a_i \leq 10^9$).

Output

The only line of the output must contain an integer: the maximal amount of fun Oleg can get.

Examples

standard input	standard output
7 1 3 2 7 3 2 4	7
7 -3 -4 -2 -2 -6 -8 -1	-11

Problem B. Oleg and Data Science

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Nowadays, everybody has heard about machine learning, neural networks and big data. And so has the student Oleg, who also wants to be in trend. He began to work hard at analyzing various datasets with Python. It's so nice to come to co-working, add a couple of layers to the neural network and, leaning back in the chair, sip the smoothies while the computer processes gigabytes of data! But today, something went wrong, and Oleg asks for your help.

Initially, he had an array a containing very important data: all integers from L to R, inclusive. Then Oleg wrote a function $f(a, m)$ which returns a new array where each integer is replaced by its remainder modulo m. Finally, Oleg mistyped and executed the line $a = f(a, Q)$, thereby replacing the original array a! To assess the scale of the tragedy, he wants to calculate the number of such positive integers X that, regardless of the content of the original array a, the result of the function $f(a, X)$ will be the same as if Oleg had not executed that hapless line.

If the problem is not clear yet, here is the mathematical statement. It is required to calculate the number of such positive integers X that, for all integers S from the segment $[L, R]$, it is true that

$$((S \bmod Q) \bmod X) = (S \bmod X).$$

Input

The only line contains three integers separated by spaces: L, R and Q ($1 \le L, R, Q \le 10^{12}$, $L \le R$).

Output

If the number of suitable positive integers X is finite, print it. Otherwise, print the word "`infinity`".

Examples

standard input	standard output
`1 1 1`	`1`
`3 5 2`	`2`
`1 10 10`	`4`
`1 1 2`	`infinity`

Problem C. Christmas Garland

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Once upon a time, Nikita was relaxing at home and watching a Christmas garland. The light bulbs were flickering following some strange pattern.

Let us formalize the garland's description. It consists of n colored light bulbs. Every light bulb is either on or off at any moment. Initially, all of them are off.

Sometimes all light bulbs of one color change their states to the opposite. After each such change, Nikita wants to know the number of maximal non-empty continuous segments of lit bulbs. A lit segment is maximal if it is not contained in any other lit segment.

Input

The first line contains integers n, k and q: the number of light bulbs, the number of different colors and the number of changes of the garland ($1 \le n, q \le 2 \cdot 10^5$, $1 \le k \le n$).

The second line contains n integers $c_1, c_2, \ldots, c_n$: the colors of light bulbs in the garland ($1 \le c_i \le k$).

Next q lines describe changes of the garland in the order they happened. Each of these lines contains an integer d_i, the color of light bulbs which have just changed their states ($1 \le d_i \le k$).

Output

The output must contain q lines. The i-th line must contain one integer: the number of maximal continuous segments of lit bulbs after i-th change.

Example

standard input	standard output
3 2 5 1 2 1 1 2 1 2 2	2 1 1 0 1

Problem E. Octopus

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Vlad is engaged in underwater photography. Most of all he likes photographing squids and octopuses. For example, did you know that an octopus has three hearts? And the fact that they can change the shape and color of the body? And here is one more astonishing... Oh, come on, Vlad, get away from the keyboard, and enough to write about the octopuses!

Vlad is not only a photographer, but also an excellent programmer. He developed a special program for automatic octopus recognition in photographs. The program receives a photograph as the input. And as the output, it gives a representation of octopus in the form of an *undirected graph*. Informally, an undirected graph is a set of points and a set of edges connecting them. The graph obtained by recognizing an octopus image always has a specific form: three or more points form the *body*, and the remaining points are parts of *tentacles* which are attached to body points so that **no more than one** tentacle is attached to each point of the body. The body is a sequence of points in which any two adjacent points, as well as the first and the last points, are connected by an edge, and the number of edges connected to each point is either two or three (if a tentacle is attached to it). A tentacle is a sequence of points in which any two adjacent points are connected by an edge, and the number of edges connected to each point is either two or one (in each tentacle, there is exactly one point with only one edge). Additionally, in this graph, there is no more than one edge between any two points, and every edge connects two different points.

When Vlad collected all his photographs, he launched the recognition process. After several hours of waiting, he got the result. However, because of a tricky bug in the code, one extra edge was added to all the graphs! After rereading the source code of his program, Vlad realized that exactly one edge was indeed added to each graph. Moreover, the added edge connects two different points which were not previously directly connected by an edge. Help Vlad to find the extra edge, and in return, he will tell you more amazing facts about octopuses!

Input

The first line contains integers n and m separated by a space: the number of points and the number of edges in the graph ($1 \le n, m \le 10^5$).

The following m lines describe the edges of the graph. Each edge is given by two different integers separated by a space: the numbers of points connected by this edge. The points are numbered starting from one. Each pair of points occurs no more than once.

It is guaranteed that the given graph was obtained by adding exactly one edge to a graph resulting from an octopus image recognition.

Output

In a single line, output two integers separated by a space: the numbers of the points between which you need to remove the edge. You can output the numbers in any order.

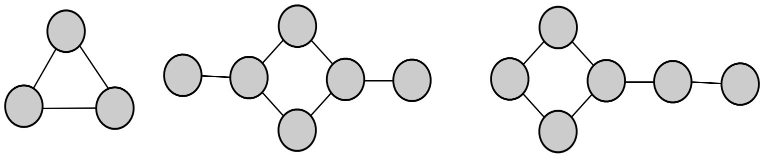
If there are several correct answers, output any one of them.

Example

standard input	standard output
7 8 1 2 2 3 3 4 4 1 3 5 5 6 5 7 3 6	3 2

Note

Here are examples of graphs which **are** correct representations of octopuses:



Here are examples of graphs which **are not** correct representations of octopuses:

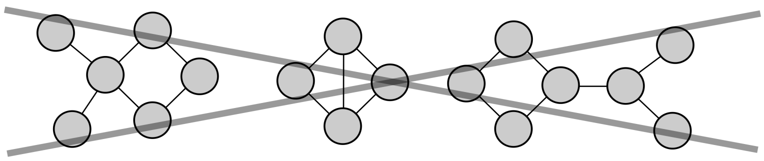
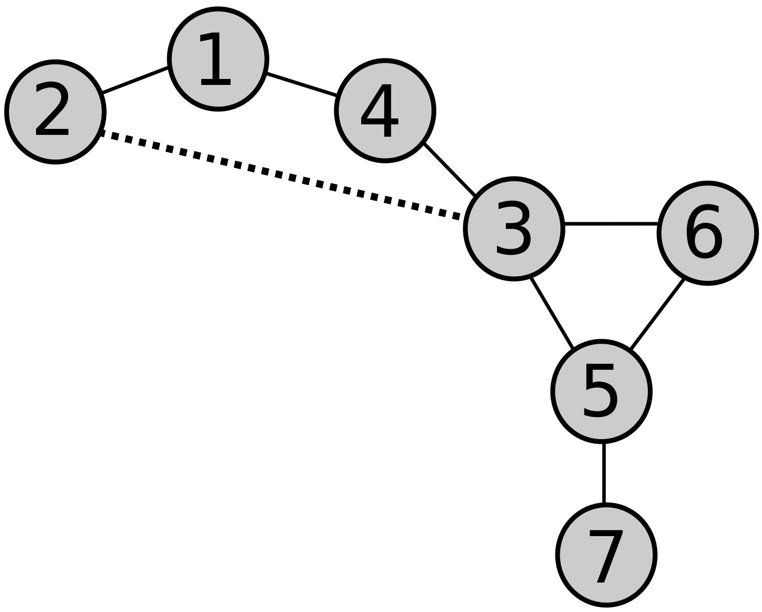


Illustration of the first test:



Problem J. Readability

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

The Martian alphabet consists of 10^9 integers from 1 to 10^9. Odd integers are vowels, and even integers are consonants. A word is a sequence of such integers.

Why do Martians have such a big alphabet? It gives the language an interesting property: changing the order of the integers in a word does not change its meaning. But, it does change its pronunciation. If each pair of adjacent integers in a word contains one vowel and one consonant, such word is called *readable*.

Vasya Pupkin decided to open a restaurant of Earthern cuisine on Mars. He bought n wooden integers for its name and placed them in a row on n pedestals, numbered from 1 to n. The integer placed on i-th pedestal is denoted as a_i.

Now Vasya wants to reorder the integers, because he wants the name of the restaurant to be readable. But moving an integer from pedestal i to pedestal j costs $|i - j|$ Martian rubles. Integers are flat, so two or more of them can be placed on one pedestal while reordering, but in the end, each pedestal must have exactly one integer on it.

Help Vasya to find a readable word formed by reordering the integers that requires him to pay as little Martian rubles as possible. If there are several such words, choose one that is the smallest lexicographically, because that will put the name of the restaurant earlier in the telephone directory.

A sequence of integers b_i is lexicographically smaller than a sequence of the same length c_i if there exists such index j that $b_i = c_i$ for all $i < j$, but $b_j < c_j$.

Input

The first line contains an integer n, the number of Martian letters in the word ($1 \le n \le 10^5$).

The second line contains n space-separated integers a_i: the letters themselves ($1 \le a_i \le 10^9$).

It is guaranteed that the solution exists.

Output

Output one line. It must contain n space-separated integers: the required readable word.

Example

standard input	standard output
5 5 3 1 4 2	5 2 1 4 3

Problem K. Robotobor

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

After the great success of Rat-O-Matic robot (see problem "I" if you like long stories), you have been promoted to CEO of Ural branch of Rapid City Dynamics company. Inspired by the local traditions, you have decided to create a very special robot called Robotobor!

Robotobor can move on a rectangular grid which consists of $n \times m$ cells. Some cells are blocked, so the robot cannot visit them. Every second, the robot moves to a cell which shares an edge with the current cell.

The movement of the robot is controlled by a program. The program consists of zero or more lines. Each line is a non-empty string of characters. Each character is one of the following: "`U`" (up), "`D`" (down), "`L`" (left) or "`R`" (right). Each character means that the robot must move to the neighboring cell in the respective direction. The program is executed line by line, every line is executed from left to right.

The program is said to be valid if and only if the following conditions hold:

- During the program execution, the robot does not visit blocked cells and does not leave the grid.
- The length of each line is at most 100 characters.
- Every line is a palindrome, that is, it reads the same right to left as left to right (Ural traditions, as wild as they are).

You want to find a valid program which moves the robot from the cell S to the cell F and consists of the minimum possible number of lines. Note that it is **not necessary** to minimize the total length of the lines. Can you repeat your previous success?

Input

The first line contains integers n and m, the number of rows and the number of columns of the grid ($1 \le n, m \le 50$).

The next n lines describe the grid. Each of these lines contains m characters. Each symbol is either "`.`" (empty cell), "`#`" (blocked cell), "`S`" (starting cell) or "`F`" (target cell). You can assume that starting and target cells are not blocked. It is guaranteed that the grid contains exactly one "`S`" and exactly one "`F`".

Output

If there is no program which satisfies all conditions, then the only line of the output must contain the number -1.

Otherwise, output the program you found. The first line must contain the number of lines k which must be the minimum possible. The next k lines must contain the program itself. If there are several possible answers, output any one of them.

Examples

standard input	standard output
3 5 S.... .#... ..#F.	2 RR DRD
2 2 F# #S	-1

Problem L. Graph Isomorphism

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Given two undirected graphs $A = (V_A, E_A)$ and $B = (V_B, E_B)$, where A's vertex set is $V_A = \{a_1, a_2, a_3, \ldots, a_{n_A}\}$, and B's vertex set is $V_B = \{b_1, b_2, b_3, \ldots, b_{n_B}\}$. Graph A and B are isomorphic if and only if

1. A and B have the same amount of vertices and edges,
2. There exists a bijective (one-to-one and onto) function $f : V_A \to V_B$ such that $\{u, v\} \in E_A$ if and only if $\{f(u), f(v)\} \in E_B$.

In other words, we can relabel the vertex set of graph A to obtain graph B.

Your task is to test if two 3-vertex undirected simple graphs G_1 and G_2 are isomorphic.

Input

The first line of the input will be a single integer T ($T \leq 100$) representing the number of test cases that will follow.

Every test case then starts with the number of edges m ($0 \leq m \leq 3$) in the first undirected simple graph of 3 vertices (numbered from 1 to 3), followed by m lines each containing two distinct integers u, v ($u \neq v$, $u, v \in \{1, 2, 3\}$) indicating that there exists an edge between vertex u and v. You may assume that there is at most one edge between any pair of vertices. After that the description of the second graph follows in the same format.

Output

If the two graphs are isomorphic than output "`yes`" on one line. If not, output "`no`" instead.

Example

standard input	standard output
3 3 1 2 2 3 3 1 3 1 3 2 1 3 2 2 1 2 1 3 0 1 2 3 1 1 2	yes no yes

Problem M. Hamming Distance

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

The Hamming distance $d_H(\vec{v}, \vec{u})$ between two n-dimensional vectors $\vec{v} = (v_1, \ldots, v_n)$ and $\vec{u} = (u_1, \ldots, u_n)$ is defined as $d_H(\vec{v}, \vec{u}) = |\{i : v_i \neq u_i \text{ and } i \in \{1, \ldots, n\}\}|$, i.e., the number of positions at which the corresponding entries are different. For example, the Hamming distance between $(1, 2, 3, 4, 5)$ and $(1, 0, 0, 4, 5)$ is 2, since these two vectors differ only at the second and the third positions. Please write a program to compute the Hamming distance between two n-dimensional vectors.

Input

On the first line there is a single integer T ($T \leq 100$) indicating the number of test cases.

Each test case consists of three lines. The first line of each test case contains an integer n ($0 < n \leq 50$) indicating the dimension of the vectors. The second line contains n integers $v_1, \ldots, v_n$, and the third line contains n integers $u_1, \ldots, u_n$. You may assume that $v_1, \ldots, v_n, u_1, \ldots, u_n \in \{0, 1, \ldots, 99\}$.

Output

For each test case, output the Hamming distance between $(v_1, \ldots, v_n)$ and $(u_1, \ldots, u_n)$.

Example

standard input	standard output
2 3 1 2 3 3 2 1 4 1 0 1 0 1 0 1 1	2 1

Problem N. X eNohpi

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 Mebibytes

¡¡X eNophi¿¿ is new smartphone from Fruit Inc. Every X eNophi has a unique serial number — integer between 1 and 10^5. Fruit Shop can sell several X eNophi with discount, if that the sum of all the serial numbers of those phones is a multiple of some positive integer M. Given all serial numbers of phones on a stock and the number M, calculate the maximum number of ¡¡X eNophi¿¿ that could have been sold from this stock with discount.

Input

Input file has the following format:

- One line with two integers N and M $(1 \le N \le 500,\ 1 \le M \le 10^5)$ — the total number of phones on the stock and the ¡¡magic¿¿ number M.
- One line with N different integers $S_1, \ldots, S_N$ $(0 \le S_i \le 10^5)$ — the serial numbers of phones on the stock.

Integers on the same line are separated by single spaces, and there will always be at least one subset of ¡¡X eNohpi¿¿ such that the sum of their serial numbers is a multiple of M.

Output

The output file should contain a single number, on a single line: the maximum number of ¡¡X eNophi¿¿ from described stock, such that the sum of their serial numbers is a multiple of M.

Example

standard input	standard output
3 5 1 8 6	3
6 9 8 6 4 1 2 3	5

Problem O. Team Composition

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Any team participating the ICPC must have three contestants, and the contestants must share one computer during the contest.

Coaches of some universities think that it is a good idea to assemble a team which has the maximum summary `Topforces` rating! Write a program to help the beginning coach to find a threshold to eliminate all teams with insufficient summary rating.

Input

The first line contains an integer T ($T \leq 100$), the number of test cases.

Each test case consists of two lines. The first line contains an integer n ($3 \leq n \leq 60$) indicating there are n candidates in the ICPC team in university. The second line contains n integers $r_1, \ldots, r_n$ separated by blanks where $r_1, \ldots, r_n \in [1, 100]$. The i-th member has `TopForces` rating r_i.

Output

For each testcase, output the maximum summary `Topforces` rating of ICPC team of this university in a line.

Example

standard input	standard output
3 3 1 2 7 6 1 2 3 4 5 6 9 77 13 22 78 54 65 7 79 99	10 15 256

Problem P. Boxing

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 Mebibytes

In amateur boxing, the two finalists are determined by a single elimination tournament. The winner of the final wins the gold, the loser wins the silver.

To qualify for bronze medal tournament, a competitor must have lost to one of the finalists.

For each case, you shall output the winner of the tournament, the runner-up and the qualifiers for bronze medal tournament.

Input

The input are a sequence of not more than 50 test cases. Each test case begins with the integer N . When N is 0, that is the signal that the input has ended and this line never should be processed. Otherwise, $2 \le N \le 6$. Next come $2^N - 1$ lines with pairs of names separated by whitespace, i.e., $name_1$, $name_2$.

Names are composed of alphabetic characters only, and will never exceed 10 characters. The meaning of each line of input is that $name_1$ defeats $name_2$.

The data always describe a perfect single elimination tournament bracket, in which half of the competitors are eliminated in round 1, half of those remaining are eliminated in round 2, and so forth, until a single winner remains. There are always exactly 2^N distinct names.

Output

For each case, output the winner of the '`Gold: `', the '`Silver: `' and the qualifiers for the '`Bronze round: `', in a space-separated, alphabetical ordered list.

Example

standard input	standard output
3 A aB CC D F E H G A CC H F A H 0	Gold: A Silver: H Bronze Round: CC F G aB